

**IN THE UNITED STATES PATENT AND TRADEMARK OFFICE**

In re Patent Application of:

Kenji ABE, et al.

Application No.:

Group Art Unit:

Filed: October 8, 2003

Examiner:

For: METHOD OF AND APPARATUS FOR VALIDATION SUPPORT, COMPUTER  
PRODUCT FOR VALIDATION SUPPORT

**SUBMISSION OF CERTIFIED COPY OF PRIOR FOREIGN  
APPLICATION IN ACCORDANCE  
WITH THE REQUIREMENTS OF 37 C.F.R. § 1.55**

Commissioner for Patents  
PO Box 1450  
Alexandria, VA 22313-1450

Sir:

In accordance with the provisions of 37 C.F.R. § 1.55, the applicant(s) submit(s)  
herewith a certified copy of the following foreign application:

Japanese Patent Application No(s). 2002-295984 and 2003-188940

Filed: October 9, 2002 and June 30, 2003

It is respectfully requested that the applicant(s) be given the benefit of the foreign filing  
date(s) as evidenced by the certified papers attached hereto, in accordance with the  
requirements of 35 U.S.C. § 119.

Respectfully submitted,

STAAS & HALSEY LLP

Date: October 8, 2003

By: 

Randall Beckers  
Registration No. 30,358

1201 New York Ave, N.W., Suite 700  
Washington, D.C. 20005  
Telephone: (202) 434-1500  
Facsimile: (202) 434-1501

日 本 国 特 許 庁  
JAPAN PATENT OFFICE

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出 願 年 月 日            2 0 0 2 年 1 0 月    9 日  
Date of Application:

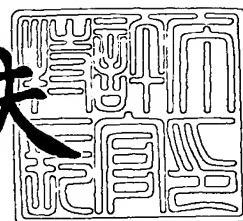
出 願 番 号            特 願 2 0 0 2 - 2 9 5 9 8 4  
Application Number:  
[ST. 10/C]:            [ J P 2 0 0 2 - 2 9 5 9 8 4 ]

出      願      人            富 士 通 株 式 会 社  
Applicant(s):

2 0 0 3 年    8 月    7 日

特許庁長官  
Commissioner,  
Japan Patent Office

今 井 康 夫



出証番号    出証特 2 0 0 3 - 3 0 6 3 4 2 3

【書類名】 特許願

【整理番号】 0252237

【提出日】 平成14年10月 9日

【あて先】 特許庁長官 殿

【国際特許分類】 G06F 11/22

【発明の名称】 検証シナリオ自動生成装置

【請求項の数】 5

【発明者】

【住所又は居所】 神奈川県川崎市中原区上小田中 4 丁目 1 番 1 号 富士通株式会社内

【氏名】 田宮 豊

【特許出願人】

【識別番号】 000005223

【氏名又は名称】 富士通株式会社

【代理人】

【識別番号】 100100930

【弁理士】

【氏名又は名称】 長澤 俊一郎

【電話番号】 03-3822-9271

【選任した代理人】

【識別番号】 100083297

【弁理士】

【氏名又は名称】 山谷 皓榮

【選任した代理人】

【識別番号】 100087848

【弁理士】

【氏名又は名称】 小笠原 吉義

## 【手数料の表示】

【予納台帳番号】 024143

【納付金額】 21,000円

## 【提出物件の目録】

【物件名】 明細書 1

【物件名】 図面 1

【物件名】 要約書 1

【包括委任状番号】 0203885

【プルーフの要否】 要

【書類名】 明細書

【発明の名称】 検証シナリオ自動生成装置

【特許請求の範囲】

【請求項 1】 一連の入出力シーケンスを被検証装置に与えて、被検証装置が正しく動作するか否かを検証するための検証シナリオを自動生成する検証シナリオ自動生成装置であって、

上記被検証装置の機能ブロック図と、リソース制約から、上記シーケンスに関する条件の論理式を生成する手段と、

上記論理式から、実行可能なシーケンスを表すすべての組み合わせのシナリオ関数を構築する手段と、

上記シナリオ関数から、検証シナリオを抽出する手段を備えたことを特徴とする検証シナリオ自動生成装置。

【請求項 2】 二分決定グラフを用いて、シナリオ関数を表現し、二分決定グラフから、検証シナリオを抽出する手段を備えたことを特徴とする請求項 1 の検証シナリオ自動生成装置。

【請求項 3】 被検証部分をしほり込む論理式を上記論理式に加えて、検証シナリオの個数を削減することを特徴とする請求項 1，2 または請求項 3 の検証シナリオ自動生成装置。

【請求項 4】 上記論理式の変数に重みを付けて、検証シナリオを定義し、予め定められた個数分の検証シナリオをコスト順に抽出する手段を備えたことを特徴とする請求項 1，2 または請求項 3 の検証シナリオ自動生成装置。

【請求項 5】 一連の入出力シーケンスを被検証装置に与えて、被検証装置が正しく動作するか否かを検証するための検証シナリオを自動生成する検証シナリオ生成プログラムであって、

上記プログラムは、上記被検証装置の機能ブロック図と、リソース制約から、上記シーケンスに関する条件の論理式を生成する処理と、

上記論理式から、実行可能なシーケンスを表すすべての組み合わせのシナリオ関数を構築する処理と、

上記シナリオ関数から、検証シナリオを抽出する処理をコンピュータに実行さ

せる

ことを特徴とする検証シナリオ自動生成プログラム。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】

通常のLSI設計において、LSIが正しく動作するかどうかの検証は、テストパターンと呼ばれる一連の入力信号シーケンスをLSIに与え、出力信号のシーケンスが期待値と合っているかどうかで行われる。

本発明は、上記テストパターンと呼ばれる入力および出力信号のシーケンスを作る元になる、検証シナリオを自動生成する検証シナリオ生成装置に関する。

上記検証シナリオから、被検証装置におけるデータの流れ（シーケンス）を得ることができ、このシーケンスに基づき、テストパターンを流すことによりLSI等の被検証機器をテストすることができる。

【0002】

【従来の技術】

被検証対象となるLSI等が正しく動作するかどうかを検証するには、上記のように、テストパターンと呼ばれる入力信号シーケンスを与えて、被検証対象が期待値通りに動作するかどうかを検証している。

しかしながら、近年、LSIの高機能化が進んでいるため、上記テストパタンの生成作業が複雑になっている。

第1の問題は、LSI内の機能ブロックの個数が非常に多いことにある。このような状況では、人手によって、全ての機能ブロック間のデータの流れ（シーケンス）を列挙する事は困難であることである。

第2の問題は、ハードウェアの並列実行性にある。ハードウェアでは、各々の機能ブロックが並列に動作する。この事は複数のシーケンスが同時に実行されることを意味している。

従来では、第1の問題を解決するため、機能ブロック図からシーケンスを抽出してきた。例えば、対象回路が、機能ブロックと、機能ブロック間のデータの流れで表されているとする。機能ブロックをノード、データの流れをエッジに置き

換えたグラフを考えると、グラフ上の 1 つのパスは、1 つのシーケンスに相当している。このグラフのパスを辿ることによって、シーケンスを列挙していた（非特許文献 1）。

#### 【0 0 0 3】

##### 【非特許文献 1】

J.Ryser, M. Glinz, "A Scenario-Based Approach to Validating and Testing Software Systems Using Statecharts", 12th Intl. Conf. on Software Engineering and their Applications, Dec. 1999.

#### 【0 0 0 4】

##### 【発明が解決しようとする課題】

上記非特許文献 1 に記載される従来の手法では、パスを 1 本ずつしか辿れないため、先に挙げた第 2 の問題を考慮できない。すなわち、複数のシーケンスが同時に実行される場合を考慮することができない。

さらに、ハードウェア特有のリソース制約（機能ブロックの共有や排他利用）も扱うことができない。

本発明は上記問題点を解決するためになされたものであって、機能ブロック図とリソース制約から実行可能な全シーケンスを抽出することができ、また、複数のシーケンスの同時実行を考慮した全ての組み合わせの検証シナリオ抽出することができる検証シナリオ生成装置を提供することである。

#### 【0 0 0 5】

##### 【課題を解決するための手段】

本発明では、図 1 に示すように、機能ブロック図とリソース制約を入力とし、以下のようにして、検証シナリオを生成する。

被検証装置の機能ブロック図と、リソース制約から、上記シーケンスに関する条件の論理式を生成し、この論理式から、実行可能なシーケンスを表すすべての組み合わせのシナリオ関数を構築し、該シナリオ関数から、検証シナリオを抽出する。

また、上記において、二分決定グラフを用いて、シナリオ関数を表現し、二分決定グラフから、検証シナリオを抽出してもよい。

なお、検証シナリオの個数を削減する場合には、被検証部分をしばり込む論理式を上記論理式に加えて検証シナリオの個数を削減したり、上記論理式の変数に重みを付けて、検証シナリオを定義し、予め定められた個数分の検証シナリオをコスト順に抽出することにより検証シナリオを削減することができる。

#### 【0006】

上記論理式は、機能ブロック図とリソース制約より自動的に生成が可能であり、生成された論理式からシナリオ関数を得て、これから自動的に検証シナリオを生成することができる。

また、二分決定グラフを用いれば、大規模なLSIのシナリオ関数であっても、実用的な計算機メモリと時間で検証シナリオを生成することが可能となる。

さらに、シーケンスの条件を論理式で表すことにより、従来技術で不可能だった、複数のシーケンスの組合せ、およびリソース制約の考慮も可能になる。

また、被検証部分をしばり込む論理式を上記論理式に加えたり、予め定められた個数分の検証シナリオをコスト順に抽出することにより、検証シナリオの数を削減することができ、検証シナリオの数が多すぎて、全てを列挙するのが現実的でない場合にも対応することができる。検証シナリオの総数は、シナリオ関数を二分決定グラフで表現し、「1-ノード」のパスを辿ることにより計算することができる。

#### 【0007】

##### 【発明の実施の形態】

図1は本発明の実施例の検証シナリオ生成装置の構成を示す図である。

図1において、1はグラフ作成部であり、LSI等の被検証装置の機能ブロックを、ノード、データの流れをエッジに置き換えたグラフに変換する。2は論理式生成部であり、上記グラフと、ハードウェア特有のリソース制約から、論理式を生成する。その際、被検証装置における被検証部分をしばり込む条件式を追加することで、生成される検証シナリオの個数を削減することができる。

3は上記生成された全ての論理式の論理積を取ったシナリオ関数であり、検証シナリオ抽出部4は、上記シナリオ関数について、値が1になるようなブール変数の組を求めることにより検証シナリオを抽出する。ここで、後述するようにシ



ナリオ関数を二分決定グラフ (Binary Decision Diagram、以下 BDD という) で表現すれば、シナリオ関数が複雑な論理式であっても、コンパクトに表現することができ、BDD の「1-ノード」へのパスを辿ることで、大規模な LSI のシナリオ関数であっても、実用的な計算機メモリと時間で、検証シナリオを抽出することが可能である。

なお、上記検証シナリオ生成装置は、CPU とメモリと外部記憶装置と入出力装置等から構成された情報処理装置により実現することができ、上記機能ブロックのグラフへの変換、論理式の生成、シナリオ関数から検証シナリオの抽出処理は、上記情報処理装置で実行されるソフトウェアにより実現することができる。

#### 【0008】

以下、図 2 に示す機能ブロック図で表される DVD/HDD ビデオレコーダの検証シナリオを抽出する場合について本発明の実施例を説明する。

図 2 は、MPEG デコーダである decMPEG と、MPEG エンコーダである encMPEG を備え、DVD-R/W、ハードディスク、ビデオからデータが入力され、DVD-R/W へのデータ出力、ハードディスクへのデータ出力、ビデオへのデータ出力を行う DVD/HDD ビデオレコーダの機能ブロック図を示している。

以下、図 2 に示した DVD/HDD ビデオレコーダの検証シナリオの生成について説明する。

##### 1. 機能ブロック図からグラフを生成する。

図 1 に示すグラフ作成部 1 は上記図 2 の機能ブロック図が与えられると、機能ブロック図における機能素子 (入力、出力、decMPEG, encMPEG) をノードとして、それらの間に流れるデータをエッジとして、グラフに変換する。そして、各機能素子とエッジに対して、ブール変数を与える。

#### 【0009】

図 3 は上記機能ブロック図から作成されたグラフであり、図 3 の例では、機能素子として以下の 8 つが定義されている [ ' ( ) ' 内は対応するブール変数 ] 。

- ・ビデオ入力 (inVideo)
- ・DVD-R/W からのデータ読み込み (inDVD)

- ・ハードディスクからのデータ読み込み (inHDD )
- ・M P E G デコーダ (decMPEG )
- ・M P E G エンコーダ (encMPEG )
- ・ビデオ出力 (outVideo)
- ・D V D - R / W へのデータ書き込み (outDVD)
- ・ハードディスクへのデータ書き込み (outHDD)

**【 0 0 1 0 】**

また、図 3 の機能素子間には、以下のようなデータが流れる [ ' ( ) ' 内は対応するブール変数] 。

- ・ビデオ入力からのビデオデータを、そのままビデオ出力へ出す (inVideo \_\_outVideo)
- ・ビデオ入力からのビデオデータを、M P E G エンコーダに渡す (inVideo \_\_encMPEG )
- ・D V D - R / W からの M P E G データを、M P E G デコーダに渡す (inDVD \_\_decMPEG )
- ・ハードディスクからの M P E G データを、M P E G デコーダに渡す (inHDD \_\_decMPEG )
- ・M P E G データをビデオデータにデコードして、ビデオ出力へ出す (decMPEG \_\_outVideo)
- ・ビデオデータを M P E G にエンコードして、D V D - R / W へ書き出す (encMPEG \_\_outDVD)
- ・ビデオデータを M P E G にエンコードして、ハードディスクへ書き出す (encMPEG \_\_outHDD)

なお、図 3 に示すグラフは、図 2 の機能ブロック図を UML ( U n i f i e d M o d e l i n g L a n g u a g e ) におけるユースケース図やシーケンス図として表現することにより、これらの図から、コンピュータにより自動的に生成することが可能である。

**【 0 0 1 1 】**

2. 論理式を生成し、シナリオ関数を生成する。

図1に示す論理式生成部2は、上記グラフから以下の(i)～(v)のように論理式を生成し、これらの論理式から以下の(vii)のようにシナリオ関数を生成する。なお、検証部分を絞り込む場合には、以下の(vi)のように検証部分を絞り込む論理式を追加する。

(i) シーケンスの始点と終点の条件を論理式に変換する。

ここで、シーケンスの始まりとなるノードを「始点」、シーケンスの終りとなる機能素子を「終点」と名付ける。

シーケンスを成立させるために、始点になっている機能素子のブール変数の論理和が1になることが必要である。同様に、終点になっている機能素子のブール変数の論理和も1になることが必要である。

図3の例では、始点と終点の条件は、それぞれ、以下の論理式(1)，(2)となる。

$\text{inVideo} + \text{inDVD} + \text{inHDD} \cdots (1)$

$\text{outVideo} + \text{outDVD} + \text{outHDD} \cdots (2)$

なお、ブール代数における、否定、論理和、論理積、含意、同値を、それぞれ、 $!$ ， $+$ ， $*$ ， $\rightarrow$ ， $\equiv$ で表している（「 $A \rightarrow B$ 」は「 $!A + B$ 」と、「 $A \equiv B$ 」は「 $(A \rightarrow B) * (B \rightarrow A)$ 」とそれぞれ同じ意味を表している）。

【0012】

(ii) 各エッジの接続条件を論理式に変換する。

各エッジについて、そのエッジ上にデータが流れれば、エッジの両端の機能素子が作動している必要がある。この条件を論理式で表現する。

図3のエッジ  $\text{inVideo} \rightarrow \text{outVideo}$ については、以下の論理式(3)式となる。 $\text{inVideo} \rightarrow \text{outVideo} \rightarrow \text{inVideo} * \text{outVideo} \cdots (3)$

その他のエッジについても同様の論理式を作ると、以下の(4)～(9)式となる。

$\text{inVideo} \rightarrow \text{encMPEG} \rightarrow \text{inVideo} * \text{encMPEG} \cdots (4)$

$\text{inDVD} \rightarrow \text{decMPEG} \rightarrow \text{inDVD} * \text{decMPEG} \cdots (5)$

$\text{inHDD} \rightarrow \text{decMPEG} \rightarrow \text{inHDD} * \text{decMPEG} \cdots (6)$

$\text{decMPEG} \rightarrow \text{outVideo} \rightarrow \text{decMPEG} * \text{outVideo} \cdots (7)$

$\text{encMPEG} \_ \text{outDVD} \rightarrow \text{encMPEG} * \text{outDVD} \dots (8)$

$\text{encMPEG} \_ \text{outHDD} \rightarrow \text{encMPEG} * \text{outHDD} \dots (9)$

### 【0013】

(iii) 各機能素子のファンアウト条件を論理式に変換する。

ある機能素子が作動すれば、そのファンアウトエッジのうち、いずれかを通してデータが次の機能素子に伝達する。反対に、ある機能素子のファンアウトエッジのどれかがデータを伝達している時は、その機能素子が作動していなくてはならない。このような条件を論理式で表す。

図3のinVideo は、ファンアウトエッジとしてinVideo  $\_ \text{outVideo}$ とinVideo  $\_ \text{encMPEG}$  を持っているので以下の論理式(10)となる。

$\text{inVideo} \equiv \text{inVideo} \_ \text{outVideo} + \text{inVideo} \_ \text{encMPEG} \dots (10)$

同様に、他の機能ブロックのファンアウトについても以下(11)～(14)式の論理式を作ることができる。

$\text{inDVD} \equiv \text{inDVD} \_ \text{decMPEG} \dots (11)$

$\text{inHDD} \equiv \text{inHDD} \_ \text{decMPEG} \dots (12)$

$\text{decMPEG} \equiv \text{decMPEG} \_ \text{outVideo} \dots (13)$

$\text{encMPEG} \equiv \text{encMPEG} \_ \text{outDVD} + \text{encMPEG} \_ \text{outHDD} \dots (14)$

### 【0014】

(iv) 各機能ブロックのファンイン条件を論理式に変換する。

この条件は、先のファンアウト条件と同様である。すなわち、ある機能素子が作動すれば、そのファンインエッジのうち、いずれかを通してデータが前の機能素子から伝達してくる。反対に、ある機能素子のファンインエッジのどれかがデータを伝達している時は、その機能素子が作動していなくてはならない。このような条件を論理式で表す。

図3のdecMPEG は、ファンインエッジとしてinDVD  $\_ \text{decMPEG}$  とinHDD  $\_ \text{decMPEG}$  を持っているので以下の(15)式の論理式となる。

$\text{decMPEG} \equiv \text{inDVD} \_ \text{decMPEG} + \text{inHDD} \_ \text{decMPEG} \dots (15)$

同様に、他の機能素子のファンインについても、以下の(16)～(19)式の論理式を作ることができる。

$\text{encMPEG} \equiv \text{inVideo} \text{ \_\_encMPEG } \cdots (16)$

$\text{outVideo} \equiv \text{inVideo} \text{ \_\_outVideo} + \text{decMPEG} \text{ \_\_outVideo} \cdots (17)$

$\text{outDVD} \equiv \text{encMPEG} \text{ \_\_outDVD} \cdots (18)$

$\text{outHDD} \equiv \text{encMPEG} \text{ \_\_outHDD} \cdots (19)$

### 【 0 0 1 5 】

(v) リソース制約を論理式に変換する。

複数のシーケンスを同時に実行しようとしても、リソースの制約から不可能な場合が考えられる。例えば、ある機能素子が複数のファンインを持っていたとしても、機能素子内部のリソース制約から、全ての入力を同時処理できないというケースが考えられる。

ここでは、そのようなハードウェアのリソース制約に起因する条件を論理式として生成する。

図 3 の、機能素子 decMPEG と outVideo について、入力を高々 1 つしか受け付けられない場合、以下の (20) ~ (21) 式の論理式となる。

$! (\text{inDVD} \text{ \_\_decMPEG } * \text{inHDD} \text{ \_\_decMPEG } ) \cdots (20)$

$! (\text{inVideo} \text{ \_\_outVideo} * \text{decMPEG} \text{ \_\_outVideo} ) \cdots (21)$

例えば上記 (21) 式は、上記リソース制約がある場合、inDVD \\_\\_decMPEG または inHDD \\_\\_decMPEG のいずれか一方が 0 でなければならないことを意味している。

また、DVD-R/W の動作速度が遅いため、データの読み込みと書き出しを同時に行えないとすると、以下の (22) 式の論理式となる。すなわち、下記論理式は DVD-R/W の読み込みと、書き出しを同時に行えないことを意味している。

$! (\text{inDVD} * \text{outDVD} ) \cdots (22)$

### 【 0 0 1 6 】

(vi) 必要に応じて、被検証部分をしばり込む論理式を追加する。

例えば、被検証対象となる L S I の内、すでに検証済の部分がある等、特定部分のみを検証すればよい場合には、被検証部分を絞り込む論理式を追加する。

例えば、図 3 において、MPEG エンコーダが使われるシナリオだけが必要だと設計者が判断する場合、「encMPEG  $\equiv 1$ 」を制約に加えれば良い。

## 【0 0 1 7】

(vii) シナリオ関数を構築する。

図 3 の例では、論理式(1) ～(22)が全て「1」にならないと、これらの論理式の全ての論理積を取ったものが、シナリオ関数となる。

大規模な L S I の場合、このシナリオ関数は非常に複雑な論理式になる可能性がある。そのような場合は、シナリオ関数を、B D D で表すのが望ましい。B D D とは、例えば図 4 に示すようなグラフを使って論理関数をコンパクトに表現するデータ構造である。B D D については、必要があれば、以下の文献を参照されたい。

R. E. Bryant, “Graph-based Algorithms for Boolean Function Manipulation”, IEEE Trans. on Computers, C-35, August 1986.

## 【0 0 1 8】

### 3. 検証シナリオを抽出する。

以上のようにしてシナリオ関数が構築されると、図 1 に示す検証シナリオ抽出部 5 は、以下のようにして、検証シナリオを抽出する。

検証シナリオの抽出は、先に構築したシナリオ関数について、値が 1 になるようなブール変数の組を列挙することである。すなわち、シナリオ関数の変数の値を、0 または 1 に変えて、条件を満たす変数の全ての組み合わせを抽出すればよい。

また、シナリオ関数が非常に複雑な論理式になる場合には、シナリオ関数を B D D で表現し、「1 ノード」へのパスを辿る事で検証シナリオを抽出することかできる。

## 【0 0 1 9】

図 3 の例では、以下の(a) ～(m) の 1 3 個の検証シナリオが抽出された。

(a) inVideo inDVD decMPEG encMPEG outVideo outHDD inVideo\_\_encMPEG inDVD \_\_decMPEG decMPEG \_\_outVideo encMPEG\_\_ outHDD

(b) inVideo inHDD decMPEG encMPEG outVideo outDVD outHDD inVideo \_\_ encMPEG inHDD\_\_ decMPEG decMPEG\_\_outVideo encMPEG\_\_ outDVD encMPEG \_\_outHDD

(c) inVideo inHDD decMPEG encMPEG outVideo outDVD inVideo\_\_encMPEG inHDD

\_\_decMPEG decMPEG \_\_outVideo encMPEG\_\_ outDVD

(d) inVideo inHDD decMPEG encMPEG outVideo outHDD inVideo\_\_encMPEG inHDD

\_\_decMPEG decMPEG \_\_outVideo encMPEG\_\_ outHDD

(e) inVideo encMPEG outVideo outDVD outHDD inVideo \_\_ outVideo inVideo \_\_

encMPEG encMPEG \_\_outDVD encMPEG\_\_outHDD

(f) inVideo encMPEG outVideo outDVD inVideo\_\_outVideo inVideo\_\_ encMPEG e  
ncMPEG\_\_ outDVD

(g) inVideo encMPEG outVideo outHDD inVideo\_\_outVideo inVideo\_\_ encMPEG e  
ncMPEG\_\_ outHDD

(h) inVideo encMPEG outDVD outHDD inVideo\_\_encMPEG encMPEG \_\_outDVD encMP  
EG\_\_ outHDD

(i) inVideo encMPEG outDVD inVideo \_\_ encMPEG encMPEG\_\_ outDVD

(j) inVideo encMPEG outHDD inVideo \_\_ encMPEG encMPEG\_\_ outHDD

(k) inVideo outVideo inVideo \_\_ outVideo

(l) inDVD decMPEG outVideo inDVD \_\_ decMPEG decMPEG\_\_ outVideo

(m) inHDD decMPEG outVideo inHDD \_\_ decMPEG decMPEG\_\_ outVideo

#### 【 0 0 2 0 】

上記検証シナリオの内、例えば(a) は、図 3 に示したグラフにおける以下の 2  
つのパスを示している。

(1) inVideo → inVideo \_\_encMPEG → encMPEG → encMPEG \_\_ outHDD → outHDD

(2) inDVD → inDVD \_\_decMPEG → decMPEG → decMPEG\_\_outVideo → outVideo

上記のように検証シナリオからパスを抽出し、このパスから実行可能なシーケ  
ンスを得る。具体的には、各パスに沿ってデータが流れるような制御コマンドを  
作成し、この制御コマンドによりテストデータを流して L S I 等の被検証機器を  
テストすればよい。

#### 【 0 0 2 1 】

また、上記シナリオ関数を B D D で表現すると、図 5 ～図 7 に示すようになる  
。なお、図 5 ～図 7 は上記シナリオ関数を図 4 に示したグラフで表すと、大きな  
グラフになることから表の形式で示したものであり、同図で「0-エッジ側ノー

ド」とは、ブール変数の値が「0」のときに辿るエッジの先のノードを指し、「1-エッジ側ノード」とは、ブール変数の値が「1」のときに辿るエッジの先のノードを指す。また、同図の `val__1` , `val__2` はそれぞれ定数0および1ノードを表す。

上記BDDにおいて、「1-ノード」へのパスを辿ることにより、前記(a)～(m)の検証シナリオを抽出することができる。

#### 【0022】

ところで、シナリオ関数が複雑な場合は、検証シナリオの総数が多すぎて、全てを列挙するのは現実的ではない場合が考えられる。

検証シナリオの総数を求めるには、シナリオ関数をBDDで表現し、「1-ノード」へのパスを辿り、パス数を数えることにより、検証シナリオの総数を計算することが可能である。

このようにシナリオ関数をBDDで表現すれば、検証シナリオを列挙する必要はなく、高速に検証シナリオの数を計算できる。したがって、検証シナリオを削減するかどうかを設計者が判断する際に、検証シナリオの総数を用いることが可能となる。

#### 【0023】

上記のように検証シナリオの数を数えた結果、検証シナリオの数が多すぎる場合には、以下のようにして検証シナリオの数を削減することができる。

(i) 検証シナリオを削減する1つ目の方法は、前記2.(vi)で説明したように、被検証部分をしばり込む論理式を追加することである。

これは、設計者の判断で重要ではない検証シナリオを除外する事に相当し、例えば図3において、MP EGエンコーダが使われるシナリオだけが必要だと設計者が判断する場合には、前記したように、「`encMPEG ≡ 1`」を制約に加えれば良い。

(ii) 検証シナリオを削減する2つ目の方法は、検証シナリオに優先順位をつけて、予め設定した個数nだけの検証シナリオを求めるようにする事である。

このためには、例えば、ブール変数に重みを導入して、検証シナリオのコストを、検証シナリオに含まれるブール変数の重み和として定義する。そして、以下



の文献で紹介されている手法を使うことによって、コスト順に  $n$  個の検証シナリオを生成する。

「島田剛一他編集，” アルゴリズム辞典”，” 最短経路（グラフ）” の項，pp .279-280，共立出版，1994 」

#### 【0 0 2 4】

図 8 は、上記のように検証シナリオの数を調べて、所望の数の検証シナリオを生成する場合の処理を説明する図であり、以下のようにして検証シナリオを生成する。

(i) 前記したように機能ブロック図からグラフを作成し、リソース制約、検証対象範囲の制限の条件を入れて論理式を生成し、シナリオ関数を求め、これに基づき、BDD を作成する。

(ii) 生成した BDD について、ブール変数に重みを導入して、検証シナリオのコストを、検証シナリオに含まれるブール変数の重み和として定義し、検証シナリオの制限個数  $n$  が与えられたら、上記文献に記載される手法を用いて、コスト順に  $n$  個の検証シナリオを生成する。

#### 【0 0 2 5】

##### 【発明の効果】

以上説明したように本発明においては、以下の効果を得ることができる。

(1) 機能ブロック図とリソース制約より、実行可能な全てのシーケンスを自動的に抽出することができる。

また、二分決定グラフを用いれば、大規模な L S I のシナリオ関数であっても、実用的な計算機メモリと時間で検証シナリオを生成することが可能となる。

(2) シーケンスの条件を論理式で表すことにより、従来技術で不可能だった、複数のシーケンスの組合せ、およびリソース制約の考慮も可能になる。

(3) 被検証部分をしほり込む論理式を上記論理式に加えたり、予め定められた個数分の検証シナリオをコスト順に抽出することにより、検証シナリオの数を削減することができ、検証シナリオの数が多すぎて、全てを列挙するのが現実的でない場合にも対応することができる。

##### 【図面の簡単な説明】

**【図 1】**

本発明の実施例の検証シナリオ生成装置の構成を示す図である。

**【図 2】**

本発明の実施例における被検証装置（DVD/HDDビデオレコーダ）の機能ブロック図である。

**【図 3】**

図 2 の機能ブロック図から作成されるグラフを示す図である。

**【図 4】**

二分決定グラフ（BDD）の一例を示す図である。

**【図 5】**

図 3 のグラフから生成されたシナリオ関数を BDD で表現した場合のノードと 0 \_\_エッジ側ノードと 1 \_\_エッジ側ノードを示す図（1）である。

**【図 6】**

図 3 のグラフから生成されたシナリオ関数を BDD で表現した場合のノードと 0 \_\_エッジ側ノードと 1 \_\_エッジ側ノードを示す図（2）である。

**【図 7】**

図 3 のグラフから生成されたシナリオ関数を BDD で表現した場合のノードと 0 \_\_エッジ側ノードと 1 \_\_エッジ側ノードを示す図（3）である。

**【図 8】**

検証シナリオの数を調べて、所望の数の検証シナリオを生成する場合の処理を説明する図である。

**【符号の説明】**

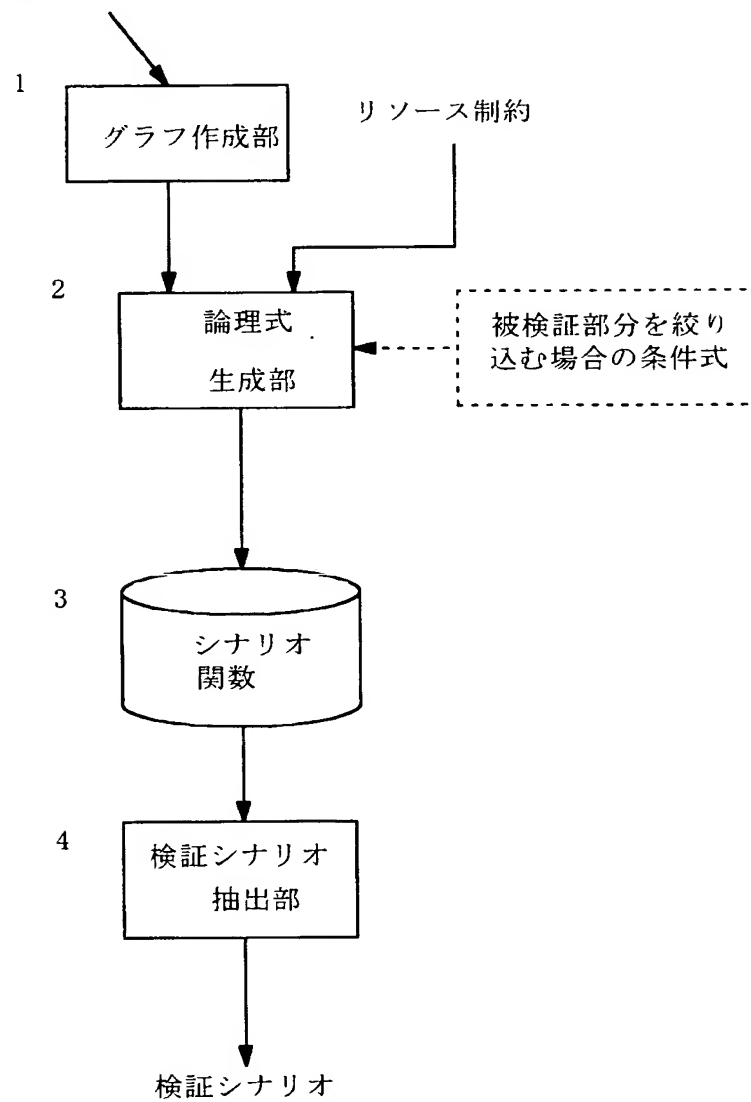
- 1     グラフ作成部
- 2     論理式生成部
- 3     シナリオ関数
- 4     検証シナリオ抽出部

【書類名】 図面

【図 1】

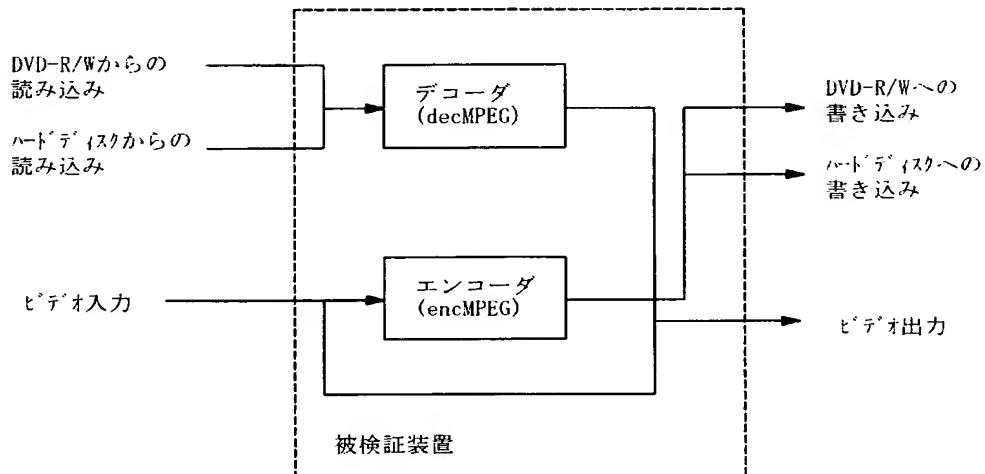
本発明の実施例の検証シナリオ生成装置の構成を示す図

機能ブロック図



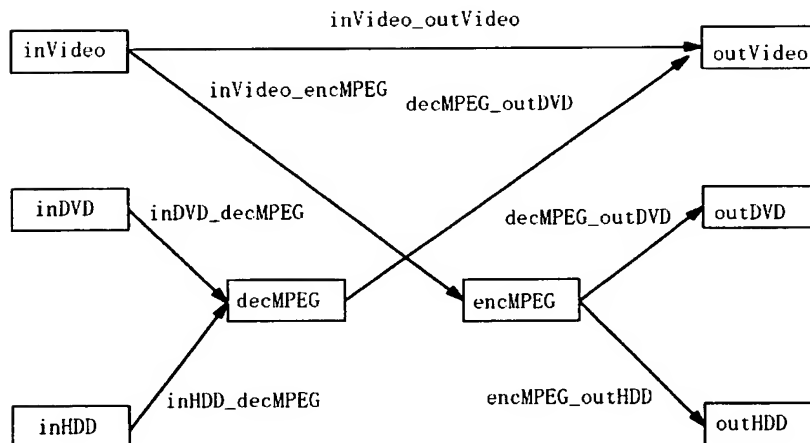
【図 2】

本発明の実施例における被検証装置（DVD/HDDビデオレコーダ）の機能ブロック図



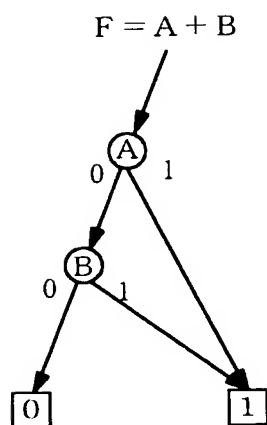
【図 3】

図 2 の機能ブロック図から作成されるグラフを示す図



【図 4】

二分決定グラフ (BDD) の一例を示す図



【図 5】

図 3 のグラフから生成されたシナリオ関数を BDD で表現した場合の  
ノードと 0\_ エッジ側ノードと 1\_ エッジ側ノードを示す図 (1)

ノード名	ブール変数	0_ エッジ側ノード	1_ エッジ側ノード
node_1	inVideo	node_2	node_26
node_2	inDVD	node_3	node_16
node_3	inHDD	val_0	node_4
node_4	decMPEG	val_0	node_5
node_5	encMPEG	node_6	val_0
node_6	outVideo	val_0	node_7
node_7	outDVD	node_8	val_0
node_8	outHDD	node_9	val_0
node_9	inVideo_outVideo	node_10	val_0
node_10	inVideo_encMPEG	node_11	val_0
node_11	inDVD_decMPEG	node_12	val_0
node_12	inHDD_decMPEG	val_0	node_13
node_13	decMPEG_outVideo	val_0	node_14
node_14	encMPEG_outDVD	node_15	val_0
node_15	encMPEG_outHDD	val_1	val_0
node_16	inHDD	node_17	val_0
node_17	decMPEG	val_0	node_18
node_18	encMPEG	node_19	val_0
node_19	outVideo	val_0	node_20
node_20	outDVD	node_21	val_0
node_21	outHDD	node_22	val_0
node_22	inVideo_outVideo	node_23	val_0
node_23	inVideo_encMPEG	node_24	val_0
node_24	inDVD_decMPEG	val_0	node_25
node_25	inHDD_decMPEG	node_13	val_0
node_26	inDVD	node_27	node_87
node_27	inHDD	node_28	node_66
node_28	decMPEG	node_29	val_0
node_29	encMPEG	node_30	node_38
node_30	outVideo	val_0	node_31
node_31	outDVD	node_32	val_0
node_32	outHDD	node_33	val_0
node_33	inVideo_outVideo	val_0	node_34
node_34	inVideo_encMPEG	node_35	val_0
node_35	inDVD_decMPEG	node_36	val_0

【図 6】

図 3 のグラフから生成されたシナリオ関数を BDD で表現した場合の  
ノードと 0\_ エッジ側ノードと 1\_ エッジ側ノードを示す図 (2)

ノード名	ブール変数	0-エッジ側ノード	1-エッジ側ノード
node_36	inHDD_decMPEG	node_37	val_0
node_37	decMPEG_outVideo	node_14	val_0
node_38	outVideo	node_39	node_60
node_39	outDVD	node_40	node_47
node_40	outHDD	val_0	node_41
node_41	inVideo_outVideo	node_42	val_0
node_42	inVideo_encMPEG	val_0	node_43
node_43	inDVD_decMPEG	node_44	val_0
node_44	inHDD_decMPEG	node_45	val_0
node_45	decMPEG_outVideo	node_46	val_0
node_46	encMPEG_outDVD	node_97	val_0
node_47	outHDD	node_48	node_54
node_48	inVideo_outVideo	node_49	val_0
node_49	inVideo_encMPEG	val_0	node_50
node_50	inDVD_decMPEG	node_51	val_0
node_51	inHDD_decMPEG	node_52	val_0
node_52	decMPEG_outVideo	node_53	val_0
node_53	encMPEG_outDVD	val_0	node_15
node_54	inVideo_outVideo	node_55	val_0
node_55	inVideo_encMPEG	val_0	node_56
node_56	inDVD_decMPEG	node_57	val_0
node_57	inHDD_decMPEG	node_58	val_0
node_58	decMPEG_outVideo	node_59	val_0
node_59	encMPEG_outDVD	val_0	node_97
node_60	outDVD	node_61	node63
node_61	outHDD	val_0	node_62
node_62	inVideo_outVideo	val_0	node_42
node_63	outHDD	node64	node_65
node_64	inVideo_outVideo	val_0	node_49
node_65	inVideo_outVideo	val_0	node_55
node_66	decMPEG	val_0	node_67
node_67	encMPEG	val_0	node_68
node_68	outVideo	val_0	node_69
node_69	outDVD	node_70	node_76
node_70	outHDD	val_0	node_71
node_71	inVideo_outVideo	node_72	val_0

## 【図 7】

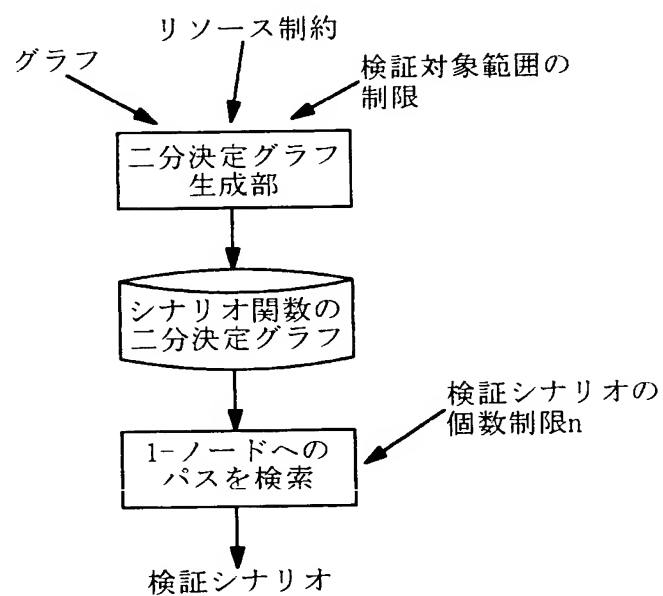
図 3 のグラフから生成されたシナリオ関数を BDD で表現した場合の  
ノードと 0\_エッジ側ノードと 1\_エッジ側ノードを示す図 (3)

ノード名	ブール変数	0-エッジ側ノード	1-エッジ側ノード
node_72	inVideo_encMPEG	val_0	node_73
node_73	inDVD_decMPEG	node_74	val_0
node_74	inHDD_decMPEG	val_0	node_75
node_75	decMPEG_outVideo	val_0	node_46
node_76	outHDD	node_77	node_82
node_77	inVideo_outVideo	node_78	val_0
node_78	inVideo_encMPEG	val_0	node_79
node_79	inDVD_decMPEG	node_80	val_0
node_80	inHDD_decMPEG	val_0	node_81
node_81	decMPEG_outVideo	val_0	node_53
node_82	inVideo_outVideo	node_83	val_0
node_83	inVideo_encMPEG	val_0	node_84
node_84	inDVD_decMPEG	node_85	val_0
node_85	inHDD_decMPEG	val_0	node_86
node_86	decMPEG_outVideo	val_0	node_59
node_87	inHDD	node_88	val_0
node_88	decMPEG	val_0	node_89
node_89	encMPEG	val_0	node_90
node_90	outVideo	val_0	node_91
node_91	outDVD	node_92	val_0
node_92	outHDD	val_0	node_93
node_93	inVideo_outVideo	node_94	val_0
node_94	inVideo_encMPEG	val_0	node_95
node_95	inDVD_decMPEG	val_0	node_96
node_96	inHDD_decMPEG	node_75	val_0
node_97	encMPEG_outHDD	val_0	val_1



【図 8】

検証シナリオの数を調べて、所望の数の検証シナリオを生成する場合の処理を説明する図



【書類名】 要約書

【要約】

【課題】 機能ブロック図とリソース制約から実行可能な全シーケンスを抽出するとともに、複数のシーケンスの同時実行を考慮した全ての組み合わせの検証シナリオ抽出すること。

【解決手段】 グラフ作成部 1 で L S I 等の被検証装置の機能ブロックを、ノード、データの流れをエッジに置き換えたグラフに変換し、このグラフと、ハードウェア特有のリソース制約から論理式生成部 3 で論理式を生成する。3 は生成された全ての論理式の論理積を取ったシナリオ関数であり、検証シナリオ抽出部 4 は、シナリオ関数 4 について、値が 1 になるようなブール変数の組を求めることで、検証シナリオを抽出する。なお、シナリオ関数を二分決定グラフで表現すれば、シナリオ関数が複雑な論理式であっても、コンパクトに表現することができ、実用的な計算機メモリと時間で、検証シナリオを抽出することができる。

【選択図】 図 1

特願 2 0 0 2 - 2 9 5 9 8 4

出 願 人 履 歴 情 報

識別番号

[ 0 0 0 0 0 5 2 2 3 ]

1 . 変更年月日

1 9 9 0 年 8 月 2 4 日

[変更理由]

新規登録

住 所

神奈川県川崎市中原区上小田中 1 0 1 5 番地

氏 名

富士通株式会社

2 . 変更年月日

1 9 9 6 年 3 月 2 6 日

[変更理由]

住所変更

住 所

神奈川県川崎市中原区上小田中 4 丁目 1 番 1 号

氏 名

富士通株式会社